# Parametric & Non-Parametric models

## Difference b/w parametric & non-parametric models

Parametric methods involve a two-step model-based approach.
1. First, we make an assumption about the functional form, or shape, of f . For example, one very simple assumption is that f is linear in X:
$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$ . (2.4)
This is a linear model, which will be discussed extensively in Chapter 3. Once we have assumed that f is linear, the problem of estimating f is greatly simplified. Instead of having to estimate an entirely arbitrary p-dimensional function f (X), one only needs to estimate the p + 1 coefficients $\beta_0, \beta_1, \ldots, \beta_p$ .
2. After a model has been selected, we need a procedure that uses the training data to fit or train the model. In the case of the linear model (2.4), we need to estimate the parameters $\beta_0, \beta_1, \ldots, \beta_p$ . That is, we want to find values of these parameters such that fit train
$Y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$ .

Non-parametric methods do not make explicit assumptions about the func-
tional form of f . Instead they seek an estimate of f that gets as close to the
data points as possible without being too rough or wiggly

| Parametric Models | Non-parametric Models |
|---|---|
| Linear regression | KNN |
| Logistic regression | |

## Advantages/disadvantages of Parametric/Non-Parametric models

Such approaches can have a major advantage over parametric approaches: by avoiding the assumption of a particular functional form for f , they have the potential to accurately fit a wider range of possible shapes for f . Any parametric approach brings with it the possibility that the functional form used to estimate f is very different from the true f , in which case the resulting model will not fit the data well. In contrast, non-parametric approaches completely avoid this danger, since essentially no assumption about the form of f is made. But non-parametric approaches do suffer from a major disadvantage: since they do not reduce the problem of estimating f to a small number of parameters, a very large number of observations (far more than is typically needed for a parametric approach) is required in order to obtain an accurate estimate for f .

# Bias & Variance

## What do we mean by the variance and bias of a statistical learning method?

A. 1. Variance refers to the amount by which $\hat{f}$ would change if we estimated it using a different training data set. Since the training data are used to fit the statistical learning method, different training data sets will result in a different $\hat{f}$ . But ideally the estimate for f should not vary too much between training sets. However, if a method has high variance then small changes in the training data can result in large changes in $\hat{f}$ . In general, more flexible statistical methods have higher variance.
On the other hand, bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model. For example, linear regression assumes that there is a linear relationship between Y and $X_1, X_2, \ldots, X_p$ . It is unlikely that any real-life problem truly has such a simple linear relationship, and so performing linear regression will undoubtedly result in some bias in the estimate of f

# SVM

## How does SVM select support vectors?

## Variations of SVM

1. Maximal Margin Classifier
2. Support Vector Classifier
3. Support Vector Machines

## What is Hyperplane?

[Source ISLR, Page 68] In a p-dimensional space, a hyperplane is a flat affine subspace of hyperplane dimension p −1 For instance, in two dimensions, a hyperplane is a flat one-dimensional subspace—in other words, a line. In three dimensions, a hyperplane is a flat two-dimensional subspace—that is, a plane. In p > 3 dimensions, it can be hard to visualize a hyperplane, but the notion of a
(p − 1)-dimensional flat subspace still applies. The mathematical definition of a hyperplane is quite simple. In two dimensions, a hyperplane is defined by the equation

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

In p-dimensional setting,

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

If a point X = (X 1 , X2, . . . , Xp )$^T$ in p-dimensional space (i.e. a vector of length p) satisfies above eq., then X lies on the hyperplane.
Now, suppose that X does not satisfy the eq; rather,

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0.$$

Then, X lies to one side of the hyperplane
On the other hand, If,

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0,$$

Then, X lies to the other side of the hyperplane.
Some Resource on equation of line
https://math.stackexchange.com/questions/2533114/equation-of-a-hyperplane-in-two-dimensions

# Linear Regression

## Assumptions of LR

1. Random error ε has E(ε) = 0
2. ε is independent of X

## Implication of the assumption that errors are independent & identically distributed

Because of this assumption, we average squared errors uniformly in our Expected Prediction error criterion. If the errors were dependent, then, weightage of each error might have been different in the error function.

## What is confidence interval and prediction interval in Linear Regression?

## How to assess quality of Linear Regression model?

A.2. [Source ISLR, Page 68]. The quality of a linear regression fit is typically assessed using two related quantities: the residual standard error (RSE) and the $R^2$ statistic.

$$\mathrm{RSE} = \sqrt{\frac{1}{n-2}\mathrm{RSS}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \qquad (3.15)$$

where,

$$\mathrm{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \qquad (3.16)$$

The RSE provides an absolute measure of lack of fit of the model (3.5) to the data. But since it is measured in the units of Y , it is not always clear what constitutes a good RSE. The $R^2$ statistic provides an alternative measure of fit. It takes the form of a proportion—the proportion of variance explained—and so it always takes on a value between 0 and 1, and is independent of the scale of Y.

To calculate $R^2$ , we use the formula,

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}} \qquad (3.17)$$

where TSS = $(y_i - \bar{y})^2$ is the total sum of squares, and RSS is defined in (3.16). TSS measures the total variance in the response Y , and can be thought of as the amount of variability inherent in the response before the regression is performed. In contrast, RSS measures the amount of variability that is left unexplained after performing the regression. Hence, TSS − RSS measures the amount of variability in the response that is explained (or removed) by performing the regression, and $R^2$ measures the proportion of variability in Y that can be explained using X.

## Derive equations for Least squares in vector & matrix notation

## Can we use Linear Regression for binary classification?

## Regression approaches in order of linearity

Source: [Source ISLR, Page 266]

| | |
|---|---|
| Linear Regression | |
| Polynomial Regression | Polynomial regression extends the linear model by adding extra predictors, obtained by raising each of the original predictors to a power. |
| Step Functions | Step functions cut the range of a variable into K distinct regions in order to produce a qualitative variable. This has the effect of fitting a piecewise constant function. |
| Regression Splines | Regression splines are more flexible than polynomials and step functions, and in fact are an extension of the two. They involve dividing the range of X into K distinct regions. Within each region, a polynomial function is fit to the data. |
| Smoothing Splines | Smoothing splines are similar to regression splines, but arise in a slightly different situation. Smoothing splines result from minimizing a residual sum of squares criterion subject to a smoothness penalty. |

| | |
|---|---|
| Local Regression | Local regression is similar to splines, but differs in an important way. The regions are allowed to overlap, and indeed they do so in a very smooth way. |
| Generalized Additive Models | Generalized additive models allow us to extend the methods above to deal with multiple predictors. |

# KNN

## Effect of K on training error

[Source: ESLR, Page 15] Error on the training data should be approximately an increasing function of k, and will always be 0 for k = 1.
It appears that k-nearest-neighbor fits have a single parameter, the number of neighbors k, compared to the p parameters in least-squares fits. Although this is the case, we will see that the effective number of parameters of k-nearest neighbors is N/k and is generally bigger than p, and decreases with increasing k. To get an idea of why, note that if the neighborhoods were nonoverlapping, there would be N/k neighborhoods and we would fit one parameter (a mean) in each neighborhood. It is also clear that we cannot use sum-of-squared errors on the training set as a criterion for picking k, since we would always pick k = 1.

## How does bias & variance vary for KNN with the choice of K?

A.4. [Source: ISLR Page 40] The choice of K has a drastic effect on the KNN classifier obtained. When K = 1, the decision boundary is overly flexible and finds patterns in the data that don't correspond to the Bayes decision boundary. This corresponds to a classifier that has low bias but very high variance. As K grows, the method becomes less flexible and produces a decision boundary that is close to linear. This corresponds to a low-variance but high-bias classifier.
Just as in the regression setting, there is not a strong relationship between the training error rate and the test error rate. With K = 1, the KNN training error rate is 0, but the test error rate may be quite high. In general, as we use more flexible classification methods, the training error rate will decline but the test error rate may not.

# Logistic Regression

## Write the Logistic Function

## Log odds

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X. \qquad (4.4)$$

## How to fit logistic regression model?

A.2. Although we could use (non-linear) least squares to fit the logistic model , the more general method of maximum likelihood is preferred, since it has better statistical properties.

## What happens when the classes are well separated in Logistic Regression?

A.3. When the classes are well-separated, the parameter estimates for the logistic regression model are surprisingly unstable. Linear discriminant analysis does not suffer from this problem.
https://stats.stackexchange.com/questions/224863/understanding-complete-separation-for-logistic-regression
https://stats.stackexchange.com/questions/239928/is-there-any-intuitive-explanation-of-why-logistic-regression-will-not-work-for

# Lasso & Ridge Regression

## Difference between Ridge & Lasso regression.

Ridge regression:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \text{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2, \qquad (6.5)$$

Lasso Regression:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|. \qquad (6.7)$$

| S.No. | Ridge Regression | Lasso Regression |
|---|---|---|
| 1 | the shrinkage penalty is applied to β1, . . . , βp , but not to the intercept β 0 . we do not want to shrink the intercept, which is simply a measure of the mean value of the response<br>when xi1 = xi2 = . . . = xip = 0. | |
| | it is best to apply ridge regression after standardizing the predictors, using the formula<br>$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\overline{x}_j)^2}},$ | |
| | ridge regression<br>will include all p predictors in the final model. The penalty λ βj 2 in (6.5)<br>will shrink all of the coefficients towards zero, but it will not set any of them<br>exactly to zero (unless λ = ∞). This may not be a problem for prediction<br>accuracy, but it can create a challenge in model | |

| | | |
|---|---|---|
| | interpretation in settings in which the number of variables p is quite large. | |
| | Uses l2 penalty | the lasso uses an l1 (pronounced “ell 1”) penalty |
| | The l2 norm of a coefficient vector β is given by $\|\|\beta\|\|_2 = \sum \beta_j^2$ | The l1 norm of a coefficient vector β is given by $\|\|\beta\|\|_1 = \sum \|\beta_j\|$. |
| | | in the case of the lasso, the 1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. |
| | | the lasso performs variable selection. |
| | | lasso yields sparse models |
| | it produces less interpretable models that involve all the predictors. | it produces simpler and more interpretable models that involve only a subset of the predictors. |

## Why is it that the lasso, unlike ridge regression, results in coefficient estimates that are exactly equal to zero?

A.2. Refer to page 221 of Introduction to Statistical Learning. Section- “*The Variable Selection Property of the Lasso*”

# Decision Trees

## Gini Index

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

Here pˆmk represents the proportion of training observations in the mth region that are from the kth class. a measure of total variance across the K classes. It is not hard to see that the Gini index takes on a small value if all of the pˆmk 's are close to zero or  one. For this reason the Gini index is referred to as a measure of node purity—a small value  indicates that a node contains predominantly observations from a single class.

## Cross-Entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$

Since $0 \le \hat{p}_{mk} \le 1$, it follows that $0 \le -\hat{p}_{mk} \log \hat{p}_{mk}$ . One can show that the cross-entropy will take on a value near zero if the pˆmk 's are all near zero or near one. Therefore, like the Gini index, the cross-entropy will take on a small value if the mth node is pure. In fact, it turns out that the Gini index and the cross-entropy are quite similar numerically.

## Why Bagging reduces over-fitting or variance?

[Source: ISLR Page 316] Given a set of n independent observations Z1 , . . . , Zn , each with variance $\sigma^2$ , the variance of the mean $\bar{Z}$ of the observations is given by $\sigma^2$ /n. In other words, averaging a set of observations reduces variance.
Explanation of why the above happens: https://en.wikipedia.org/wiki/Variance#Properties
Hence a natural way to reduce the variance and hence increase the prediction accuracy of a statistical learning method is to take many training sets from the population, build a separate prediction model using each training set, and average the resulting predictions. In other words,

we could calculate $f^{\hat{}1}$ (x), $f^{\hat{}2}$ (x), . . . , $f^{\hat{}B}$ (x) using B separate training sets, and average them in order to obtain a single low-variance statistical learning model, given by

$$\hat{f}_{\text{avg}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^b(x).$$

Of course, this is not practical because we generally do not have access to multiple training sets. Instead, we can bootstrap, by taking repeated samples from the (single) training data set. In this approach we generate B different bootstrapped training data sets. We then train our method on the bth bootstrapped training set in order to get $f^{*B}$ (x), and finally average all the predictions, to obtain

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x).$$

## OOB Error Estimation

[Source: ISLR Page 317] The key to bagging is that trees are repeatedly fit to bootstrapped subsets of the observations. One can show that on average, each bagged tree makes use of around two-thirds of the observations. The remaining one-third of the observations not used to fit a given bagged tree are referred to as the out-of-bag (OOB) observations. We can predict the response for the ith observation using each of the trees in which that observation was OOB. This will yield around B/3 predictions for the ith observation. In order to obtain a single prediction for the ith observation, we can average these predicted responses (if regression is the goal) or can take a  majority vote (if classification is the goal). This leads to a single OOB prediction for the ith observation. An OOB prediction can be obtained in this way for each of the n observations, from which the overall OOB MSE (for a regression problem) or classification error (for a classification  problem) can be computed. The resulting OOB error is a valid estimate of the test error for the  bagged model, since the response for each observation is predicted using only the trees that were not fit using that observation.

## How Random Forests ensure that trees are decorrelated

[Source: ISLR Page 320] Random forests provide an improvement over bagged trees by way of a random small tweak  that decorrelates the trees. As in bagging, we build a number  forest of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh sample of √ m predictors is taken at each split, and typically we choose m ≈ p—that is, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors (4 out of the 13 for the Heart data). In other words, in building a random forest, at each split in the tree, the algorithm is not even allowed to consider a majority of the available predictors. This may sound crazy, but it has a clever rationale. Suppose that there is one very strong predictor in the data set, along with a num- ber of other moderately strong predictors. Then in the collection of bagged trees, most or all of the trees will use this strong predictor in the top split. Consequently, all of the bagged trees will look quite similar to each other. Hence the predictions from the bagged trees will be highly correlated. Un- fortunately, averaging many highly correlated quantities does not lead to as large of a reduction in variance as averaging many uncorrelated quanti- ties. In particular, this means that bagging will not lead to a substantial reduction in variance over a single tree in this setting. Random forests overcome this problem by forcing each split to consider only a subset of the predictors. Therefore, on average (p − m)/p of the splits will not even consider the strong predictor, and so other predictors will have more of a chance. We can think of this process as decorrelating the trees, thereby making the average of the resulting trees less variable and hence more reliable. The main difference between bagging and random forests is the choice of predictor subset size m. For instance, if a random forest is built using m = p, then this amounts simply to bagging.

## Does Random Forest overfit if we increase the number of trees

[Source: ISLR Page 321] As with bagging, random forests will not overfit if we increase B, so in practice we use a value of B sufficiently large for the error rate to have settled down.

## Does Boosting overfit if we increase the number of trees

[Source: ISLR Page 323] Unlike bagging and random forests, boosting can overfit if B is too large, although this overfitting tends to occur slowly if at all. We use cross-validation to select B. (B is the number of trees)

# Appendix

## How to sample from Normal distribution?